On the Implementation of Cryptosystems Based on Real Quadratic Number Fields
(Extended Abstract)

Detlef Hühnlein[1] and Sachar Paulus[2]

[1] Secunet Security Networks AG, Germany
huehnlein@secunet.de
[2] SAP AG, Germany
sachar.paulus@t-online.com

Abstract. Cryptosystems based on the discrete logarithm problem in the infrastructure of a real quadratic number field [7,19,2] are very interesting from a theoretical point of view, because this problem is known to be at least as hard as, and when considering todays algorithms – as in [11] – much harder than, factoring integers. However it seems that the cryptosystems sketched in [2] have not been implemented yet and consequently it is hard to evaluate the practical relevance of these systems. Furthermore as [2] lacks any proofs regarding the involved approximation precisions, it was not clear whether the second communication round, as required in [7,19], really could be avoided without substantial slowdown. In this work we will prove a bound for the necessary approximation precision of an exponentiation using quadratic numbers in power product representation and show that the precision given in [2] can be lowered considerably. As the highly space consuming power products can not be applied in environments with limited RAM, we will propose a simple (CRIAD[1]-) arithmetic which entirely avoids these power products. Beside the obvious savings in terms of space this method is also about 30% faster. Furthermore one may apply more sophisticated exponentiation techniques, which finally result in a ten-fold speedup compared to [2].

1 Introduction

Unlike for imaginary quadratic orders, there is no polynomial time algorithm known, which decides whether two given ideals in a real quadratic order $\mathcal{O}_\Delta$ are equivalent, and consequently it is impossible to set up DL-based cryptosystems in real quadratic class groups $Cl(\Delta)$. However, as noted by Shanks [21], there is some infrastructure in the cycle of reduced principal ideals, which resembles an abelian group. Buchmann, Williams and Scheidler [7,18,19] showed how to construct a Diffie-Hellman-like key agreement procedure using this infrastructure. They (essentially) represent a principal ideal $\mathfrak{A}$ by a pair $(\mathfrak{a}, a)$, where $\mathfrak{a} = \gamma\mathfrak{A}$ is

[1] CRIAD is an abbreviation for Close Reduced Ideal and Approximated relative Distance

D.R. Stinson and S. Tavares (Eds.): SAC 2000, LNCS 2012, pp. 288–302, 2001.

the reduced ideal *closest* to $\mathfrak{A}$ and a is a rational approximation to the distance $\log(\gamma)$ between $\mathfrak{a}$ and $\mathfrak{A}$. The major problem with their approach is that, because it is (in general) impossible to find the closest reduced ideal when using a fixed approximation precision, they compute a pair of very close reduced ideals, i.e. the left and right neighbour, and uniquely determine the common key in a *second communication round*. Because this additional round is necessary in their setup, it is impossible to construct more advanced – e.g. signature- – protocols. In [2] it was proposed to use the exact relative generator $\gamma \in \mathbb{Q}(\sqrt{\Delta})$ in power product representation [6] and only compute a rational approximation of its logarithm in the very end. The claimed that it is possible to avoid the second communication round and hence implement advanced cryptographic protocols (c.f. [3]). However as [2] lacks any proofs regarding the involved approximation precisions it was not clear whether the second communication round, as required in [7,19], can be avoided at all, and if, without substantially decreasing the efficiency.

In this work we will prove a bound for the necessary approximation precision of an exponentiation using quadratic numbers in power product representation and show that the precision given in [2] may be lowered considerably. For a discriminant with 1024 bit and 160 bit secret exponents in a Diffie-Hellman key-agreement protocol we will see that a precision of $512 + 2 + 160 + 3 = 677$ bits is sufficient, where [2] suggest a precision of $1024 + 6 \cdot 160 + 6 = 1990$ bits for this scenario. As in more sophisticated cryptographic protocols it is necessary to have, not only an exponentiation but also, a multiplication (and possibly inversion) routine available, we will introduce the so called CRIAD-multiplication, which is (essentially) associative (see Corollary 1). Using CRIADmult one is able to construct a binary exponentiation routine which requires a precision of $512 + 2 + 2 \cdot 160 + 2 = 836$ bits in the above scenario, but *entirely avoids power products* and solely uses floating point numbers for the logarithms. Note that this is very important in environments with restricted RAM. This binary exponentiation variant (CRIADexp Algorithm 3) with a precision of 836 bits already yields a running time, which is about 30% faster than the exponentiation variant using power products and an approximation precision of 677 bits. Another important feature of this approach is that one may use CRIADmult to implement more sophisticated exponentiation routines, which finally result in a ten-fold speedup compared to [2].

The paper is organized as follows: In Section 2 we will recall the necessary basics concerning the infrastructure of the principal class in a real quadratic number field. In Section 3 we carry together the necessary definitions concerning rational approximations of real numbers. In Section 4 we will recall the representation of principal ideals from [2] and prove lower bounds for the required approximation precision to provide a unique representation (see Proposition 2), which is necessary to avoid the second communication round, as required in [7,19]. In Section 5 we will show what precision is necessary to implement an exponentiation technique using power products. In Section 6 we will introduce the CRIAD-arithmetic which is necessary to implement more sophisticated protocols and allows to avoid the application of power products. In Section 7 we will use this basic arithmetic to derive more sophisticated exponentiation techniques. Due to space restrictions we will only give the results and refer to [12] for a detailed presentation and proofs. We will conclude this work by providing timings of a first implementation in Section 8. In the full paper [12] we will also provide a complexity analysis and the treatment of more sophisticated cryptographic – e.g. signature – protocols.

2 The Infrastructure of the Principal Class in Real Quadratic Number Fields

In this section we will recall the very basic notions of the infrastructure of the principal class in real quadratic number fields needed in the sequel and refer to [8,10,1,13] for a complete reference.

Let $\mathbb{Q}(\sqrt{\Delta})$ be the real quadratic number field of discriminant $\Delta > 0$ and $\mathcal{O}_\Delta$ be its ring of integers. We denote $\mathcal{O}_\Delta$-ideals by gothic letters $\mathfrak{a}, \mathfrak{b}, \ldots, \mathfrak{A}, \mathfrak{B}, \ldots$. An important invariant of the number field $\mathbb{Q}(\sqrt{\Delta})$ is the regulator $\mathcal{R}_\Delta = \log \varepsilon$, where ε is the smallest unit larger than one. It is well known (see e.g. [20]), that the computation of $\mathcal{R}_\Delta$ is at least as hard as factoring Δ.

Two $\mathcal{O}_\Delta$-ideals $\mathfrak{a}$ and $\mathfrak{b}$ are called *equivalent* if there is a $\gamma \in \mathbb{Q}(\sqrt{\Delta})$ such that $\mathfrak{a}\gamma = \mathfrak{b}$. γ is called a *relative generator* of $\mathfrak{b}$ w.r.t. $\mathfrak{a}$. γ is unique up to multiplication by units. Equivalence of ideals is an equivalence relation. The equivalence classes are called *ideal classes.* If $\mathfrak{a} = \mathcal{O}_\Delta$ then $\mathfrak{b} = \gamma\mathcal{O}_\Delta$ is called a *principal ideal* and γ is simply called a *generator* of $\mathfrak{b}$. The set of principal ideals is denoted by $\mathcal{P}_\Delta$ and forms an infinite abelian subgroup of $\mathcal{I}_\Delta$. The factor group $Cl(\Delta) = \mathcal{I}_\Delta/\mathcal{P}_\Delta$ is a finite abelian group and is called the *ideal class group* of $\mathbb{Q}(\sqrt{\Delta})$.

For two equivalent ideals $\mathfrak{a}$ and $\mathfrak{b} = \gamma\mathfrak{a}$ we define the *distance* between $\mathfrak{a}$ and $\mathfrak{b}$ by

$$\delta(\mathfrak{a}, \mathfrak{b}) = \frac{1}{2} \log \left| \frac{\gamma}{\overline{\gamma}} \right| \pmod{\mathcal{R}_\Delta}, \tag{1}$$

where $\overline{\gamma} = (x - y\sqrt{\Delta})/z$ denotes the (real) conjugate of $\gamma = (x + y\sqrt{\Delta})/z$. If $\mathfrak{a} = \mathcal{O}_\Delta$, we simply write $\delta(\mathfrak{b})$ instead of $\delta(\mathcal{O}_\Delta, \mathfrak{b})$.

Given an ideal $\mathfrak{A}$ we denote by $\rho()$ (see e.g. [13, REDUCE_REAL, Algorithm 2.6]) the reduction operator, which computes an equivalent reduced ideal $\mathfrak{a} = \rho(\mathfrak{A})$. We denote $(n \geq 1)$ successive applications of $\rho()$ by $\rho^n()$. If $\mathfrak{a}$ is a reduced ideal then $\rho^n(\mathfrak{a})$ is also reduced and there is some $l \in \mathbb{Z}_{>1}$, such that $\mathfrak{a} = \rho^l(\mathfrak{a})$. In [22] it is shown that, for arbitrary Δ, the smallest such l may be as large as $O(\sqrt{\Delta} \log\log\Delta)$.

Let $\mathfrak{a} = (a, b)$ be a reduced ideal. Then one may use [13, Algorithm 2.11] to compute the *right neighbour* $\mathfrak{a}_+ = (a_+, b_+) = \rho(\mathfrak{a})$, where $\delta(\mathfrak{a}, \mathfrak{a}_+) = \frac{1}{2} \log \left| \frac{b_+ + \sqrt{\Delta}}{b_+ - \sqrt{\Delta}} \right|$ and [13, Algorithm 2.12] respectively to compute the *left neighbour* $\mathfrak{a}_- = (a_-, b_-) = \rho^{-1}(\mathfrak{a})$, where $\delta(\mathfrak{a}, \mathfrak{a}_-) = -\frac{1}{2} \log \left| \frac{b_- + \sqrt{\Delta}}{b_- - \sqrt{\Delta}} \right|$ of $\mathfrak{a}$.

It is easy to see that traveling around a complete circle and obtaining $\rho^l(\mathfrak{a}) = \mathfrak{a}$ and thus $\mathfrak{a}(\epsilon) = \mathfrak{a}$ yields

$$\delta(\rho^l(\mathfrak{a}), \mathfrak{a}) = \frac{1}{2} \log \left| \frac{\epsilon}{\bar{\epsilon}} \right| = \frac{1}{2} \log \left| \frac{\epsilon^2}{N(\epsilon)} \right| = \frac{1}{2} \log(\epsilon^2) = \log \epsilon = \mathcal{R}_\Delta.$$

Instead of this naive strategy, which is obviously only applicable for very small Δ, Shanks [21] made use of the *infrastructure* of the principal class of a real quadratic order to compute $\mathcal{R}_\Delta$. In the following we will recall the most important properties of this infrastructure.

In [15] it was shown that

$$\log\left(1/\sqrt{\Delta} + 1\right) < |\delta(\mathfrak{a}, \mathfrak{a}_+)| < \log \sqrt{\Delta}, \tag{2}$$

and for three consecutive ideals $\mathfrak{a}, \mathfrak{a}_+, \mathfrak{a}_{++}$ we have

$$|\delta(\mathfrak{a}, \mathfrak{a}_{++})| > \log 2. \tag{3}$$

Moreover, it is immediate from the definition that for three equivalent ideals $\mathfrak{a}, \mathfrak{b}, \mathfrak{c}$ we have

$$\delta(\mathfrak{a}, \mathfrak{b}) + \delta(\mathfrak{b}, \mathfrak{c}) \equiv \delta(\mathfrak{a}, \mathfrak{c}) \pmod{\mathcal{R}_\Delta} \tag{4}$$

and for two pairs $(\mathfrak{a}, \mathfrak{b})$ and $(\mathfrak{c}, \mathfrak{d})$ of principal ideals

$$\delta(\mathfrak{a}\mathfrak{c}, \mathfrak{b}\mathfrak{d}) \equiv \delta(\mathfrak{a}, \mathfrak{c}) + \delta(\mathfrak{b}, \mathfrak{d}) \pmod{\mathcal{R}_\Delta}. \tag{5}$$

The last assertions are not surprising, as the set of (invertible) principal ideals $\mathcal{P}_\Delta$ forms a group under multiplication. On the other hand it is easy to see that the subset of *reduced* principal ideals, together with some combination of multiplication and reduction, does not form a group, because such an operation is either not associative or not closed.

However it is possible to show that the operation $\odot$, defined by multiplication followed by reduction, is "close to" being a group operation:

Proposition 1. *Let* $\mathfrak{a}, \mathfrak{b}$ *be reduced ideals and* $\mathfrak{c} = \mathfrak{a} \odot \mathfrak{b} = \rho(\mathfrak{a}\mathfrak{b})$. *Then*

$$|\delta(\mathfrak{c}, \mathfrak{a}\mathfrak{b})| = |\delta(\mathfrak{c}) - (\delta(\mathfrak{a}) + \delta(\mathfrak{b}))| < 2 \log \Delta. \tag{6}$$

Proof. See [8, Proposition 5.8.4].

We will see that this deficiency w.r.t. a group operation can be repaired, without applying the space consuming power product representation from [6]. In the multiplication procedure for principal ideals in CRIAD-representation (CRIADmult, Algorithm 2) we will additionally compute rational approximations for distances, which allow to correct the "error" introduced by reduction. We will see in Corollary 1, that this strategy makes the operation CRIADmult (essentially) associative, if one uses a sufficiently high approximation precision.

3 Rational Appromations of Real Numbers

The distances between equivalent ideals, as defined in (1), are of the form $1/2 \log |\gamma/\overline{\gamma}|$, for $\gamma \in \mathbb{Q}(\sqrt{\Delta})^*$, and hence in general irrational numbers. As the most practical way to handle these distances seems to be the computation of – sufficiently accurate – rational approximations, we will follow [5,17] und carry together the necessary definitions and properties of *floating point numbers.*

Let $r \in \mathbb{R}$, $r \neq 0$. Then we define $b(r) = \lfloor \log_2 |r| \rfloor + 1$ and $b(0) = 0$.

Definition 1. *A* floating point number *is a pair $f = (m, e)$, where $m, e \in \mathbb{Z}$, $m \neq 0$ or $m = 0$ and $e = 0$. m is called the* mantissa *and e is called the* exponent *of f. $f = (m, e)$ represents the rational number $q = m \cdot 2^{e-b(m)}$.*

Definition 2. *Let $r \in \mathbb{R}$ and $k \in \mathbb{Z}, j \in \mathbb{Q}_{>0}$.*

1. *An* absolute k-approximation *for r is a floating point number $f = (m, e)$, such that $|f - r| < 2^{-k}$ and $e \geq b(m) - k - 1$.*
2. *An* absolute (j, k)-approximation *for r is a floating point number $f = (m, e)$, such that $|f - r| < \frac{j}{2^k}$ and $e \geq b(m) - k + \lceil log_2(j) \rceil - 1$.*

Remark 1. The latter definition is necessary to consider the round-off-error in some computation more closely. Here we will shortly relate a (j, k)-approximation to an l-approximation.

Let $f = (m, e)$, where $e \geq b(m) - k + \lceil log_2(j) \rceil - 1$, be an absolute (j, k)-approximation for $r \in \mathbb{R}$. Then $|f - r| < \frac{j}{2^k} = 2^{\log_2(\frac{j}{2^k})} = 2^{\log_2(j)-k}$ and one immediately sees that f is precisely an absolute l-approximation for $l = k - \lceil \log_2(j) \rceil$. On the other side it is clear from the definition that a $(1, k)$-approximation is precisely a k-approximation.

We use Markus Maurer's functions from the xbigfloat-class of LiDIA [16] to implement the necessary floating point arithmetic. A theoretical treatment of these functions may be found in [17]. Besides addition, subtraction and the comparison of floating point numbers we will also need the following two functions:

- $\mathsf{Trunc}(f, k)$
  denotes the LiDIA-function `Truncate`(f, k) and returns the "k significant bits" of a floating point number $f = (m, e) = m \cdot 2^{e-b(m)}$ by deleting the last $b(m) - k$ last bits of the mantissa m. To prove a bound for the necessary precision to make our proposed CRIAD-representation unique, we will make use of [17, Lemma 2.3.1], which states that given a k-approximation $f = (m, e)$ of a real number r, $f' = \mathsf{Trunc}(f, k + e)$ is a $k - 1$-approximation of r.
- $\mathsf{qlog}(x, y, k)$
  denotes the LiDIA-function a.`absolute_Ln_approximation`(k) and returns on input of a number $\gamma = x + y\sqrt{\Delta}$ a k-approximation of Lenstra's [15] logarithm $\mathrm{Log}(\gamma) = 1/2|\gamma/\overline{\gamma}|$. A thorough description of this function may be found in [17, Section 6.1.4].

4 Bounds for the Uniqueness of the CRIAD-Representation

In this section we introduce the CRIAD-representation of principal ideals, as sketched in [2]. After a formal specification of the required properties we will derive bounds for the involved precision to make this representation unique.

To make the line of thought leading to this important representation more transparent, we will start with defining a RIAD-representation, where we do not require that the reduced ideal in this representation is close to the represented ideal.

Definition 3. *Let $\mathfrak{A}$ be a (fractional) principal $\mathcal{O}_\Delta$-ideal, $f = (m, e)$ a floating point number and $k \in \mathbb{Z}, j \in \mathbb{Q}_{>0}$.*

A (j, k)-RIAD-representation of $\mathfrak{A}$ is a pair $(\mathfrak{a}, f)$, where $\mathfrak{a}$ is an arbitrary reduced principal ideal and f is an absolute (j, k)-approximation for the distance $\delta(\mathfrak{A}, \mathfrak{a})$ between $\mathfrak{A}$ and $\mathfrak{a}$. A $(1, k)$-RIAD-representation is simply called k-RIAD-representation.

Given a, not necessarily reduced, principal ideal $\mathfrak{A}$ in standard representation it is easy to compute a k-RIAD-representation $(\mathfrak{a}, f)$ for it. The procedure $(\mathfrak{a}, f) = \mathsf{Std2RIAD}(\mathfrak{A}, k)$ uses the standard LiDIA-routine $(\mathfrak{a}, \gamma) = \mathsf{REDUCE}(\mathfrak{A})$ (see e.g. [13, REDUCE_REAL, Algorithm 2.6]), which computes $\mathfrak{a} = \rho(\mathfrak{A})$ and the relative generator $\gamma = (x + y\sqrt{\Delta})/z = \mathfrak{A}/\mathfrak{a}$, and computes $f = -\mathsf{qlog}(x, y, k)$.

As the reduced ideal $\mathfrak{a}$ in this representation will, for example, be used to derive a common key in a Diffie-Hellman key agreement procedure, it is especially important to guarantee that both communication partners end up with the same reduced ideal $\mathfrak{a}$, representing $\mathfrak{A} = \mathfrak{g}^{ab}$, while performing entirely different computations.

As there are no further requirements for the reduced ideal $\mathfrak{a}$ in the RIAD-representation $(\mathfrak{a}, f)$ of a principal ideal $\mathfrak{A}$ and there are $c = O(\sqrt{\Delta} \log\log \Delta)$ reduced ideals in the principal cycle, there are obviously c different RIAD-representations for $\mathfrak{A}$. Among all these RIAD-representations for $\mathfrak{A}$ we will now elect the CRIAD-representation, which will be shown to be uniquely determined if the involved precision is suffiently high. If the precision would be too low, such that there would be two valid CRIAD-representations $(\mathfrak{a}_i, f_i)$, $i \in \{1, 2\}$, $\mathfrak{a}_1 \neq \mathfrak{a}_2$, for an ideal $\mathfrak{A} = \mathfrak{g}^{ab}$, then a key agreement procedure would entirely fail to work, or would need to be "repaired" using a second communication round, as proposed in [7,19].

Suppose for a moment, that the distances could be determined exactly. Then one could simply define the unique representative for $\mathfrak{A}$ to be the reduced ideal $\mathfrak{a}$ with the (in absolute value) smallest distance and possibly – if $\mathfrak{A}$ is *precisely* inbetween two reduced ideals – positive distance. In this case it is clear that $\mathfrak{a}$ is uniquely determined.

However, since we are dealing with rational approximations, i.e. we only have $k < \infty$ many correct bits of the distances at our disposal, some more considerations are necessary to make the reduced ideal in the CRIAD-representation unique.

Suppose, that all computations were performed with a sufficiently high precision k, that we can guarantee, that the reduced ideal $\mathfrak{a}$ in the RIAD-representation $(\mathfrak{a}, f)$ is either the left or right neighbour of $\mathfrak{A}$ and recall that the function $\mathsf{Trunc}(g, l)$ returns (only) the l significant bits of a floating point number g.

Let $(\mathfrak{a}_1, f_1)$, with $f_1 = (m_1, e_1)$, and $(\mathfrak{a}_2, f_2)$, with $f_1 = (m_1, e_1)$, be two canditates for the CRIAD-representation. If $|\mathsf{Trunc}(f_1, k + e_1)| < |\mathsf{Trunc}(f_2, k + e_2)|$, then the decision is easy and we choose $(\mathfrak{a}_1, f_1)$ to be the unique CRIAD-representation.

In the worst case $\mathfrak{A}$ is – in the scope of the fixed approximation precision – precisely inbetween the two reduced ideals $\mathfrak{a}_1$ and $\mathfrak{a}_2$. Then the RIAD-representations $(\mathfrak{a}_1, f_1)$ and $(\mathfrak{a}_2, f_2)$ have the following properties:

For the distances we have

$$|\delta(\mathfrak{A}, \mathfrak{a}_1) - f_1| < 2^{-k} \text{ and } |\delta(\mathfrak{A}, \mathfrak{a}_2) - f_2| < 2^{-k}, \tag{7}$$

since we have k-RIAD-representations and

$$\mathsf{Trunc}(f_1, k + e_1) = -\mathsf{Trunc}(f_2, k + e_2), \tag{8}$$

since $\mathfrak{A}$ is – in the scope of the fixed approximation precision – precisely inbetween $\mathfrak{a}_1$ and $\mathfrak{a}_2$.

We assume w.l.o.g. that $f_1 > 0$ and choose $(\mathfrak{a}_1, f_1)$ to be the unique CRIAD-representation for $\mathfrak{A}$, even if the precise distances may satisfy

$$|\delta(\mathfrak{A}, \mathfrak{a}_1)| > |\delta(\mathfrak{A}, \mathfrak{a}_2)|.$$

Thus the reduced ideal $\mathfrak{a}$ in the CRIAD-representation $(\mathfrak{a}, f)$ is not necessarily the reduced ideal closest to $\mathfrak{A}$, but nevertheless *uniquely determined*, if the approximation precision, as part of the system parameters, is sufficiently high.

Now we will bring the above vague ideas in a more formal shape, such that we will be able to determine the necessary precision in order to guarantee that the reduced ideal $\mathfrak{a}$ in a CRIAD-representation is uniquely determined.

Definition 4. *Let $k \in \mathbb{Z}, j \in \mathbb{Q}_{>0}$ and $l = k - \lceil \log_2(j) \rceil$. Then a (j, k)-CRIAD-representation of $\mathfrak{A}$ is defined to be a (j, k)-RIAD-representation $(\mathfrak{a}, f)$, where $f = (m, e)$, of $\mathfrak{A}$, satisfying the following properties:*

1. *$|\mathit{Trunc}(f, l+e)| \leq |\mathit{Trunc}(f', l+e')|$ for all (j, k)-RIAD-representations $(\mathfrak{a}', f')$, where $f' = (m', e')$, of $\mathfrak{A}$ and*
2. *if $(\mathfrak{a}_1, f_1)$ and $(\mathfrak{a}_2, f_2)$ are two (j, k)-RIAD-representations, which satisfy (1.), where $f_1 > 0$ and $f_2 < 0$, then $(\mathfrak{a}, f) = (\mathfrak{a}_1, f_1)$.*

If $\mathfrak{A}$ is reduced, then we call $(\mathfrak{A}, 0)$ a $(0, k)$-CRIAD-representation for any $k \in \mathbb{Z}$. A $(1, k)$-CRIAD-representation is simply called a k-CRIAD-representation.

Definition 5. *A (j, k)-CRIAD-representation $(\mathfrak{a}, f)$ is called* unique*, if there is no $l = k - \lceil \log_2(j) \rceil$-CRIAD-representation $(\mathfrak{a}', f')$, $\mathfrak{a}' \neq \mathfrak{a}$, for a given ideal $\mathfrak{A}$, which satisfies (1.) and possibly (2.) in above definition.*

It remains to determine a bound for the precision to make such a CRIAD-representation unique. For this purpose we will proceed in two steps. In Lemma 1 we will present a different formulation of the uniqueness-problem for CRIAD-representations. We will show that this representation is unique if in a certain real, open, interval of width $2^{-k+\lceil \log_2(j)\rceil+2}$ there is only one reduced ideal. In Proposition 2 we will derive a bound such that in an interval of said width there can be only one reduced ideal.

Lemma 1. *Let* $(\mathfrak{a}_1, f_1)$*, where* $f_1 = (m_1, e_1)$*, be a* (j,k)*-CRIAD-representation of a principal ideal* $\mathfrak{A}$*,* $l = k - \lceil \log_2(j)\rceil$ *and* $f = \mathsf{Trunc}(f_1, l + e_1)$*. This CRIAD-representation is unique, if there is no reduced ideal* $\mathfrak{a}_2 \neq \mathfrak{a}_1$*, such that*

$$\delta(\mathfrak{a}_2) \ \in\ ]\delta(\mathfrak{A}) - f - 2^{-l+1}, \delta(\mathfrak{A}) - f + 2^{-l+1}[.$$

Proof. Let $l = k - \lceil \log_2(j)\rceil$. Considering the above definition, we see that the (j,k)-CRIAD-representation $(\mathfrak{a}_1, f_1)$, with $f_1 = (m_1, e_1)$, for a principal ideal $\mathfrak{A}$ is not unique if there is (at least) one other (j,k)-CRIAD-representation $(\mathfrak{a}_2, f_2)$, with $f_2 = (m_2, e_2)$, for $\mathfrak{A}$, such that $\mathfrak{a}_2 \neq \mathfrak{a}_1$ and $f = \mathsf{Trunc}(f_1, l + e_1) = \mathsf{Trunc}(f_2, l + e_2)$.

Considering the involved distances, we have

$$|\delta(\mathfrak{A}, \mathfrak{a}_i) - f_i| < j/2^k \leq 2^{-l}, \quad i \in \{1, 2\},$$

as $(\mathfrak{a}_i, f_i)$ are (j,k)-CRIAD-representations.

By [17, Lemma 2.3.1] we loose one bit of precision by computing

$$f = \mathsf{Trunc}(f_1, l + e_1) = \mathsf{Trunc}(f_2, l + e_2).$$

I.e. as f_i, $i \in \{1, 2\}$, are absolute l-approximations for $\delta(\mathfrak{A}, \mathfrak{a}_i)$ we know that f is an absolute $l - 1$-approximation for $\delta(\mathfrak{A}, \mathfrak{a}_i)$ and we obtain

$$|\delta(\mathfrak{A}, \mathfrak{a}_i) - f| = |\delta(\mathfrak{A}) - f - \delta(\mathfrak{a}_i)| < 2^{-l+1}, \quad i \in \{1, 2\}.$$

This shows that non-uniqueness occurs, if there is some reduced ideal $\mathfrak{a}_2 \neq \mathfrak{a}_1$, such that

$$\delta(\mathfrak{a}_2) \ \in\ ]\delta(\mathfrak{A}) - f - 2^{-l+1}, \delta(\mathfrak{A}) - f + 2^{-l+1}[. \qquad \square$$

Looking back to our original argumentation, Lemma 1 shows that non-uniqueness occurs, if there is a reduced ideal $\mathfrak{a}'$, such that the CRIAD-representation $(\mathfrak{a}', f')$ satisfies all requirements in the definition, but $\mathfrak{a}'$ is *not the direct* left or right neighbour of $\mathfrak{A}$.

To derive a bound for the uniqueness of the CRIAD-representation, it is – by Lemma 1 – sufficient to investigate, whether in a real, open interval of width $2^{-k+\lceil \log_2(j)\rceil+2}$ there may be two (or more) reduced ideals.

Proposition 2. *Let* $(\mathfrak{a}, f)$ *be a* (j,k)*-CRIAD-representation of a principal* $\mathcal{O}_\Delta$*-ideal* $\mathfrak{A}$*,* $l = k - \lceil \log_2(j)\rceil$ *and*

$$\chi(\Delta) = -\log_2\left(\log\left(1/\sqrt{\Delta} + 1\right)\right) + 2. \tag{9}$$

Then the (j,k)*-CRIAD-representation* $(\mathfrak{a}, f)$ *of* $\mathfrak{A}$ *is* unique*, if* $l \geq \chi(\Delta)$*.*

Proof. Let $l = k - \lceil \log_2(j) \rceil$. Then – by Lemma 1 – it is sufficient to explore, whether two neighbouring, reduced ideals $\mathfrak{a}, \mathfrak{a}_+$ may lie in an open interval of width 2^{-l+2}. By (2) we have

$$|\delta(\mathfrak{a}, \mathfrak{a}_+)| > \log(1/\sqrt{\Delta} + 1). \tag{10}$$

We have uniqueness, if it is impossible that two neighbouring ideals lie in the interval of width 2^{-l+2}. By (10) we have $2^{-l+2} \leq \log(1/\sqrt{\Delta} + 1) < |\delta(\mathfrak{a}, \mathfrak{a}_+)|$ and therefore

$$l \geq -\log_2(\log(1/\sqrt{\Delta} + 1)) + 2 = \chi(\Delta).$$

□

Remark 2. During the execution of a cryptographic protocol one needs to take care that one remains above this minimum precision.

For $x > 1$ we have $\log(1/x + 1) > 1/(x + 1)$ and one obtains the bound

$$\chi(\Delta) < \log_2(\sqrt{\Delta} + 1) + 2. \tag{11}$$

Thus it is sufficient, that – at the end of any cryptographic protocol – about $\log_2(\Delta)/2$ correct bits of the distances are at our disposal. Note that the cursory "analysis" in [2] suggests a minimum precision of $\log_2(\Delta)$ bits.

5 CRIAD-Exponentiation Using Power Products

In this section we will show what precision is sufficient in an exponentiation routine for ideals in CRIAD-representation using power products, as in [2]. As above, our analysis will reveal that the precision bounds given in [2] are way too high.

We will use the LiDIA-function $(\mathfrak{b}, d) = \mathsf{CLOSE}(\mathfrak{a}, t, k)$ which on input of a reduced ideal $\mathfrak{a}$, a rational number t and an approximation precision k will return a reduced ideal $\mathfrak{b}$ such that $\delta(\mathfrak{b})$ is – with respect to k – close to $\delta(\mathfrak{a}) + t$ and a k-approximation d to $\mathrm{Log}(\mathfrak{a}/\mathfrak{b})$. A detailed description of this function can be found in [17, Section 8.4]. Note that the functionality of CLOSE is equal to the functionality of the procedure TARGET in [2].

Let $n \in \mathbb{Z}_{>0}$ and $l = \lfloor \log_2(n) \rfloor$. Then $(n_l, \cdots, n_0)$ is the binary expansion of $n = \sum_{i=0}^{l} n_i 2^i$, where $n_i \in \{0, 1\}$ for $0 \leq i \leq l - 1$, $n_l = 1$.

Proof. Recall that in any call $(\mathfrak{a}, \alpha) = \mathsf{REDUCE}(\mathfrak{A})$ we have $\mathfrak{a} = \mathfrak{A}/\alpha$.

Thus we obtain at the end of the **for**-loop $\mathfrak{a}^n = \mathfrak{h}\gamma$ and compute the floating point number t such that $\delta(\mathfrak{h}) + t$ is close to $\delta(\mathfrak{A}^n)$. Therefore the correctness, disregarding the approximation precision, follows from the correctness of CLOSE [17, Section 8.4].

It remains to show the correctness of the J-value. Let $\theta = \mathfrak{A}^n/\mathfrak{h}$. Then we have $|t - \mathrm{Log}(\theta)| < nj2^{-k} + 2^{-(k+z)} \leq (2^{l+1} - 1)j2^{-k} + 2^{-(k+z)}$ and at the very end $|\mathfrak{A}^n - (\delta(\mathfrak{h} + h)| < (2^{l+1} - 1)j2^{-k} + 3 \cdot 2^{-(k+z)}$ which shows that $J = (2^{l+1} - 1)j + 3 \cdot 2^{-z}$ is correct. □

Algorithm 1 CRIADexpPP

Require: The (j,k)-CRIAD-representation $(\mathfrak{a}, f)$ of a principal $\mathcal{O}_\Delta$-ideal $\mathfrak{A}$, the exponent $n = (n_l, \cdots, n_0) \in \mathbb{Z}_{>0}$, the final precision k and the additional precision $z \geq 0$.
Ensure: The (J,k)-CRIAD-representation $(\mathfrak{b}, g)$ of $\mathfrak{A}^n$, where $J = (2^{l+1}-1)j + 3 \cdot 2^{-z}$.

```
$\gamma = 1$
$\mathfrak{h} = \mathfrak{a}$
for $i = l-1$ downto $0$ do
    $(\mathfrak{h}, \alpha) \leftarrow \mathsf{REDUCE}(\mathfrak{h}^2)$
    $\gamma \leftarrow \gamma^2\alpha$
    if $n_i = 1$ then
        $(\mathfrak{h}, \alpha) \leftarrow \mathsf{REDUCE}(\mathfrak{h} \cdot \mathfrak{a})$
        $\gamma \leftarrow \gamma\alpha$
    end if
end for
$t \leftarrow f \cdot n - \mathsf{qlog}(\gamma, k+z)$
$(\mathfrak{h}, h) \leftarrow \mathsf{CLOSE}(\mathfrak{h}, t, k+z)$
$h \leftarrow t - h$
return$(\mathfrak{h}, h)$
```

To allow a fair comparison of the different exponentiation strategies we need to consider their behaviour within some cryptographic protocol. We will only give bounds for the Diffie-Hellman key-agreement-protocol here and treat more sophisticated protocols in the final paper [12].

Proposition 3. *Let $a, b < 2^{l+1}$ be the secret exponents in a Diffie-Hellman key-agreement protocol using CRIADexpPP and a reduced ideal as common base. Then it is sufficient if both partners use $k \geq \chi(\Delta)$ and $z \geq l+3$.*

Proof. As both partners start with a reduced ideal, i.e. a $(0,k)$-CRIAD-representation, they obtain a (j_1, k) CRIAD-representation with the first exponentiation, where $j_1 = 3 \cdot 2^{-z}$. The second exponentiation yields a (j_2, k)-representation, where $j_2 = (2^{l+1}-1)j_1 + 3 \cdot 2^{-z} = 3 \cdot 2^{l+1-z} < 2^{l+3-z}$. This is a k-approximation, if $z \geq l+3$. □

This proposition explains the required precision of $512 + 2 + 160 + 3 = 677$ bits, for 1024 bit Δ and 160 bit exponents, stated in the introduction.

6 CRIAD-Arithmetic without Power Products

Regardless of the applied exponentiation technique it is necessary to have the procedure CRIADmult – and possibly CRIADinv – available to implement more sophisticated – e.g. signature – protcols. Therefore we will develop this basic arithmetic for principal ideals in CRIAD-representation. We will show in Corollary 1, that CRIADmult is (essentially) a group operation. Thus one may conststruct exponentiation techniques based on this procedure which do not require the power product representation and consequently can be applied in environments with limited RAM.

In our presentation of CRIADmult we will need a procedure $(\mathfrak{b}, g) = \mathsf{RIAD2CRIAD}((\mathfrak{a}, f), z)$ which uses right- or leftsteps to convert a given (j, k)-RIAD-representation approximation into the $(j+1, k)$-CRIAD-representation. One may[2] use the procedure LOCAL_CLOSE [17, Section 8.2] for this purpose.

Algorithm 2 CRIADmult

Require: The (j_a, k)-CRIAD-representation $(\mathfrak{a}, a)$ of a principal ideal $\mathfrak{A}$, the (j_b, k)-CRIAD-representation of a principal ideal $\mathfrak{B}$, the final precision k and an additional precision $z \in \mathbb{Z}_{\geq 0}$, where $k \geq \chi(\Delta) + \log_2(j_a + j_b + 2^{-z})$.
Ensure: The unique $(j_a + j_b + 2^{-z}, k)$-CRIAD-representation $(\mathfrak{c}, c)$ of $\mathfrak{A}\mathfrak{B}$.

```
$p \leftarrow k + z + 1$
$(\mathfrak{h}, h) \leftarrow \mathsf{Std2RIAD}(\mathfrak{a} \cdot \mathfrak{b}, p)$
$(\mathfrak{c}, c) \leftarrow \mathsf{RIAD2CRIAD}((\mathfrak{h}, h + a + b), p)$
return$(\mathfrak{c}, c)$
```

Proof. The proof will appear in the full paper [12]. □

Now it is easy to see that this operation is (essentially) associative, provided that the approximation precision is chosen to be sufficiently large.

Corollary 1. *Let $z \geq 0$ and $(\mathfrak{a}, a), (\mathfrak{b}, b), (\mathfrak{c}, c)$ be the unique $(j_a, k), (j_b, k)$, (j_c, k)-CRIAD-representations for the principal ideals $\mathfrak{A}, \mathfrak{B}, \mathfrak{C}$, $J = j_a + j_b + j_c + 2^{-z+1}$, where $k \geq \chi(\Delta) + \lceil \log_2(J) \rceil$. Let $(\mathfrak{d}_1, d_1) = \mathit{CRIADmult}(\mathit{CRIADmult}((\mathfrak{a}, a), (\mathfrak{b}, b), z), (\mathfrak{c}, c), z)$, where $d_1 = (m_1, e_1)$ and $(\mathfrak{d}_2, d_2) = \mathit{CRIADmult}((\mathfrak{a}, a), \mathit{CRIADmult}((\mathfrak{b}, b), (\mathfrak{c}, c), z), z)$, where $d_2 = (m_2, e_2)$. Then $\mathfrak{d}_1 = \mathfrak{d}_2$ and $\mathit{Trunc}(d_1, k + e_1) = \mathit{Trunc}(d_2, k + e_2)$.*

Proof. See [12]. □

Remark 3. Since we have chosen Lenstra's distance $\mathrm{Log}(\gamma) = 1/2 \log |\gamma/\overline{\gamma}|$, as proposed in [15], instead of Shanks' naive distance $\log(\gamma)$ [21], it is easy to see that the *inversion* of a principal ideal in CRIAD-representation is essentially free of cost and especially does not impose any round-off-errors. If $((a, b), f)$ is a (j, k)-CRIAD-representation of $\mathfrak{A}$, then $((a, -b), -f) = \mathsf{CRIADinv}((a, b), f)$ is a

[2] Note that LOCAL_CLOSE makes use of (small) power products. Due to space restrictions we need to refer to the final paper [12] for our method RIAD2CRIAD, which avoids power products at the cost of a slightly higher internal precision

(j,k)-RIAD[3]-representation of $\mathfrak{A}^{-1}$. This fact will be used to construct signed digit exponentiation routines, which are slightly more efficient.

7 CRIAD-Exponentiation without Using Power Products

As CRIADmult essentially behaves like a group operation it is straightforward to construct – more sophisticated – exponentiation routines for principal ideals in CRIAD-representation.

Due to space restrictions we will only present the exponentiation routine based on the classical binary square and multiply strategy in detail here. For the more sophisticated exponentiation routines we will only present the results of the precision analysis; the corresponding proofs appear in the full paper [12].

Algorithm 3 CRIADexp

Require: The (j,k)-CRIAD-representation $(\mathfrak{a}, f)$ of a principal $\mathcal{O}_\Delta$-ideal $\mathfrak{A}$, the exponent $n = (n_l, \cdots, n_0) \in \mathbb{Z}_{>0}$, the final precision k and the additional precision $z \geq 0$.

Ensure: The (J,k)-CRIAD-representation $(\mathfrak{b}, g)$ of $\mathfrak{A}^n$, where $J = (2^{l+1}-1)j + 2^{-z}(2^{l+1}-2)$.

```
$(\mathfrak{h}, h) \leftarrow (\mathfrak{a}, f)$
for $i = l-1$ downto $0$ do
  $(\mathfrak{h}, h) \leftarrow \mathsf{CRIADmult}((\mathfrak{h}, h), (\mathfrak{h}, h), k, z)$
  if $n_i = 1$ then
    $(\mathfrak{h}, h) \leftarrow \mathsf{CRIADmult}((\mathfrak{h}, h), (\mathfrak{a}, f), k, z)$
  end if
end for
return$(\mathfrak{h}, h)$
```

Remark 4. It should be noted that the presented square and multiply strategy is the so called “left-to-right” variant. This is important, because it features *less error propagation* for reduced ideals than the “right-to-left” variant [14], while the number of group operations is the same. This is yet another point, where the precision stated in [2] can be easily improved.

Proposition 4. *Let $a, b < 2^{l+1}$ be the secret exponents in a Diffie-Hellman key-agreement protocol using CRIADexp and a reduced ideal as common base. Then it is sufficient if both partners use $k \geq \chi(\Delta)$ and $z \geq 2l + 2$.*

In a similar manner we obtain the following bounds for more sophisticated exponentiation techniques; for the proofs we need to refer to the final paper [12].

[3] It should be noted that, if the ideal is *precisely* inbetween two reduced ideals, another right step might be necessary to obtain the CRIAD-representation

Proposition 5. *On input of a (j,k)-CRIAD-representation and the exponent n with $l = \lfloor \log_2(n) \rfloor$, the sliding m-bit window method (see e.g. [9]) produces a (J,k)-CRIAD-representation, where $J = 2^{l+m+1} j + 2^{l+m+1} 2^{-z}$.*

Let $a, b < 2^{l+1}$ be the secret exponents in a Diffie-Hellman key-agreement protocol using CRIADexpwindow and a reduced ideal as common base. Then it is sufficient if both partners use $k \geq \chi(\Delta)$ and $z \geq 2(l+m)+3$.

Proposition 6. *On input of a (j,k)-CRIAD-representation, the appropriate precomputed values and the exponent n with $l = \lfloor \log_2(n) \rfloor$, the signed 2^m-digit version of the BGMW exponentiation method [4] produces a (J,k)-CRIAD-representation, where $J = 2^{l+m+2} j + 2^{l+m+3} 2^{-z}$.*

Let $a, b < 2^{l+1}$ be the secret exponents in a Diffie-Hellman key-agreement protocol using CRIADexpBGMW and a reduced ideal as common base. Then it is sufficient if both partners use $k \geq \chi(\Delta)$ and $z \geq 2(l+m)+6$.

If the first exponentiation is performed with CRIADexpBGMW and the second exponentiation is performed with CRIADexpwindow, which might be often used in practice, then it is sufficient if both partners use $k \geq \chi(\Delta)$ and $z \geq 2(l+m)+5$.

8 Timings of a First Implementation

In this section we provide timings of a first implementation using the different exponentiation methods discussed in this work. For the sake of comparison, we also provide timings of an implementation of the procedure EXP as given in [2]. The timings in Table 1 are given in seconds and correspond to randomly chosen discriminants and exponents of the respective bit length on a Pentium II with 166 MHz using LiDIA 2.0 [16]. As precision we used the necessary precision for the Diffie-Hellman protocol as given in the Propositions 3, 4, 5, 6 and [2] respectively.

The results for 500 bit discriminants and 100 bit exponents, which are closest to real world requirements, indicate that an exponentiation with CRIADexp is more than twice as fast as the exponentiation routine EXP from [2] and can – using precomputation – be accelerated to obtain a more than ten times faster exponentiation. Moreover our approach without applying power products seems to save not only a considerable amount of space, but also up to 30% time. Because the relative speedup tends to increase with higher parameters one might conjecture that our method is also asymptotically preferable. This issue will be discussed in the full paper [12].

Acknowledgement

We would like to thank Renate Scheidler, Markus Maurer and Hugh C. Williams for fruitful discussions and for making us aware of mistakes in an earlier draft of this paper.

Table 1. Timings for different CRIAD-exponentiations

Bitlength	Bitlength	Exponentiation method				
of	of	EXP	CRIADexpPP	CRIADexp	CRIADexpwindow	CRIADexpBGMW
Exponent	Discriminant	see [2]	see Prop.3	see Prop.4	see Prop.5	see Prop.6
20	100	0.50	0.38	0.33	0.44	0.11
40	100	1.10	0.94	0.88	0.93	0.28
60	100	2.14	1.82	1.64	1.71	0.39
80	100	3.52	2.85	2.75	2.68	0.60
100	100	5.71	4.18	4.12	3.79	0.82
20	200	0.99	0.77	0.60	0.66	0.17
40	200	2.09	1.70	1.37	1.49	0.27
60	200	3.68	2.91	2.47	2.47	0.55
80	200	5.66	4.23	3.90	3.68	0.72
100	200	8.68	5.99	5.66	5.21	1.10
20	300	1.76	0.87	0.83	0.99	0.27
40	300	3.35	1.92	1.93	1.81	0.44
60	300	5.33	3.35	3.18	2.91	0.82
80	300	8.13	5.00	4.83	4.23	1.04
100	300	11.53	7.91	6.76	6.37	1.48
20	400	1.76	1.48	1.10	1.32	0.32
40	400	3.79	3.07	2.36	2.59	0.55
60	400	6.64	4.89	3.96	4.06	0.88
80	400	10.10	7.25	5.99	5.88	1.20
100	400	14.88	10.00	8.73	8.35	1.70
20	500	3.35	2.58	1.38	1.43	0.43
40	500	7.14	4.61	2.97	2.91	0.66
60	500	10.88	7.30	5.06	4.61	1.15
80	500	15.98	10.27	7.75	6.96	1.65
100	500	22.52	13.84	10.44	9.66	2.09

References

1. I. Biehl and J. Buchmann: *Algorithms for Quadratic Orders.* Proceedings of Symposia in Applied Mathematics. **48**. American Mathematical Society: 1994. pp. 425-451.
2. I. Biehl, J. Buchmann and C. Thiel: *Cryptographic Protocols Based on Discrete Logarithms in Real-quadratic Orders*, Advances in Cryptology – CRYPTO '94, LNCS **839**, Springer, 1995, pp. 56 – 60
3. I. Biehl, B. Meyer and C. Thiel: *Cryptographic Protocols Based on Real-Quadratic A-fields.* Proceedings of ASIACRYPT '96. Springer: 1996. pp. 15-25.
4. E. Brickell, D. Gordon, K. McCurley, D. Wilson: *Fast Exponentiation with Precomputation*, Advances in Cryptology, EUROCRYPT '92, LNCS **658**, Springer, 1993, pp. 200-207
5. J. Buchmann, M. Maurer: *Approximate Evaluation of* $L(1, \chi_\Delta)$, Technical Report, Darmstadt, University of Technology, 1997
6. J. Buchmann, C. Thiel, H.C. Williams: *Short representation of quadratic integers*, Computational Algebra and Number Theory, Mathematics and its Applications **325**, 1995, pp. 159 – 185
7. J. Buchmann and H.C. Williams: *A Key Exchange System Based on Real Quadratic Fields.* Proceedings of CRYPTO '89. Springer: 1989. pp. 335-343.

8. H. Cohen: *A Course in Computational Algebraic Number Theory.* Graduate Texts in Mathematics **138**. Springer: Berlin, 1993.
9. H. Cohen: *Analysis of the flexible window powering algorithm*, preprint available via `http://www.math.u-bordeaux.fr/ cohen/`
10. L.K. Hua: *Introduction to Number Theory.* Springer-Verlag: New York, 1982.
11. D. Hühnlein: *Quadratic orders for NESSIE - Overview and parameter sizes of three public key families*, submitted to ISSE 2000, preprint available via `http://www.informatik.tu-darmstadt.de/TI/Veroeffentlichung/TR/Welcome.html`
12. D. Hühnlein, M. Maurer, S. Paulus: *On the complexity and efficiency of cryptosystems using real quadratic number fields*, Technical report TU Darmstadt, to appear, 2000
13. M.J. Jacobson Jr.: *Subexponential Class Group Computation in Quadratic Orders*, PhD-thesis, TU Darmstadt, appeared in Shaker, Aachen, ISBN 3-8265-6374-3, 1999
14. D.E. Knuth: *The Art of Computer Programming.* Vol. 2: Seminumerical algorithms. Addison-Wesley, Reading MA, 1981.
15. H.W. Lenstra: *On the computation of regulators and class numbers of quadratic fields*, London Math. Soc. Lecture Notes, **56**, 1982, pp. 123-150
16. LiDIA: *A c++ library for algorithmic number theory,* via http://www.informatik.tu-darmstadt.de/TI/LiDIA
17. M. Maurer: *Regulator approximation and fundamental unit computation for real quadratic orders*, PhD-thesis, TU-Darmstadt, to appear 2000
18. R. Scheidler, J. Buchmann, H.C. Williams: *Implementation of a key exchange protocol using real quadratic fields (extended abstract)*, Advances in Cryptology – EUROCRYPT '90, Springer, LNCS **473**, 1991, pp. 98 – 109
19. R. Scheidler, J. Buchmann and H.C. Williams: *A Key-Exchange Protocol Using Real Quadratic Fields.* Journal of Cryptology **7**. 1994. pp. 171-199.
20. R.J. Schoof: *Quadratic Fields and Factorization.* In: H.W. Lenstra, R. Tijdeman, (eds.): *Computational Methods in Number Theory.* Math. Centrum Tracts **155**. Part II. Amsterdam, 1983. pp. 235-286.
21. D. Shanks, *The infrastructure of a real quadratic field and its applications.* Proc. Number Theory Conference, Boulder. 1972, pp. 217-224.
22. H.C. Williams: *A numerical investigation into the length of the period of the continued fraction expansion of* $\sqrt{D}$, Math. Comp. **36**, 1981, pp. 593-601